To appear in the IEEE International Joint Conference on Biometrics, Shenzhen, August 2021

FedFace: Collaborative Learning of Face Recognition Model

Divyansh Aggarwal, Jiayu Zhou, Anil K. Jain
Michigan State University
East Lansing, MI, USA
{aggarw49, jiayuz, jain}@cse.msu.edu

Abstract

DNN-based face recognition models require large centrally aggregated face datasets for training. However, due to the growing data privacy concerns and legal restrictions, accessing and sharing face datasets has become exceedingly difficult. We propose FedFace, a federated learning (FL) framework for collaborative learning of face recognition models in a privacy aware manner. FedFace utilizes the face images available on multiple clients to learn an accurate and generalizable face recognition model where the face images stored at each client are neither shared with other clients nor the central host and each client is a mobile device containing face images pertaining to only the owner of the device (one identity per client). Our experiments show the effectiveness of FedFace in enhancing the verification performance of pre-trained face recognition system on standard face verification benchmarks namely LFW, IJB-A and IJB-C.

1. Introduction

Automated Face Recognition (AFR) systems have been widely adopted for access control and person identification in a plethora of domains largely due to their accuracy, usability, and touchless and remote acquisition. AFR systems are embedded in our smartphones to unlock our phones, facilitate financial transactions as well as secure access to stored personal information. According to the 2020 NIST face recognition evaluation (FRVT) [10], only 0.08% of searches in a database of 26.6 million face images failed to find the true mate, compared to a failure rate of 4% in 2014. That corresponds to a 50-fold improvement in face recognition performance over the last six years. This significant boost in face recognition performance can be largely attributed to the use of Deep Neural Networks (DNNs) and large face training datasets for learning salient face representations.

978-1-6654-3780-6/21/$31.00 ©2021 IEEE

DNN-based face recognition systems require large and diverse (in terms of race, gender and facial variations such as pose, illumination and expression) training face datasets to learn robust and generalizable face representations. However, due to the sensitive nature of biometric data (including face images), it is becoming exceedingly difficult for researchers to access large face datasets. According to a report by the United States Government Accountability Office (GAO), with the rapid proliferation of face images on social media websites such as Facebook, Twitter, and Instagram, face recognition models are being trained without obtaining the consent of the individuals whose face images are being used [5]. A face recognition startup, Clearview AI, is currently facing litigation for allegedly amassing a dataset of about 3 billion face images through internet scraping without acquiring consent from the individuals in the dataset [3]. Article 25 of the General Data Protection Regulation (GDPR) [6], enforces data protection "by design and by default" in the development of any new framework and strictly prohibits collection and distribution of personal data without consent. Similar statutes exist in the states of California (California Consumer Privacy Act, CCPA) [2] and Illinois (Personal Information Protection Act, PIPA) [8] to impose strict regulations on breach of personal data, including face images. Several large scale publicly available face datasets are now being retracted on account of containing face images of individuals without their approvals. Examples include Microsoft's MS Celeb [14], one of the largest face datasets with over $10M$ images from $10,000$ individuals which has been used by several commercial organizations (IBM, Alibaba, Nvidia and Sensetime) and academic research groups to train facial recognition systems. It was pulled off the internet in light of inclusion of face images of individuals without their consent [9]. Duke MTMC surveillance data set collected by Duke University researchers, and a Stanford University data set, called Brainwash, were also retracted from public domain on similar grounds [4, 1]. On account of growing and legitimate data privacy concerns on the ways training datasets for face recognition models can be collected and used, training AFR

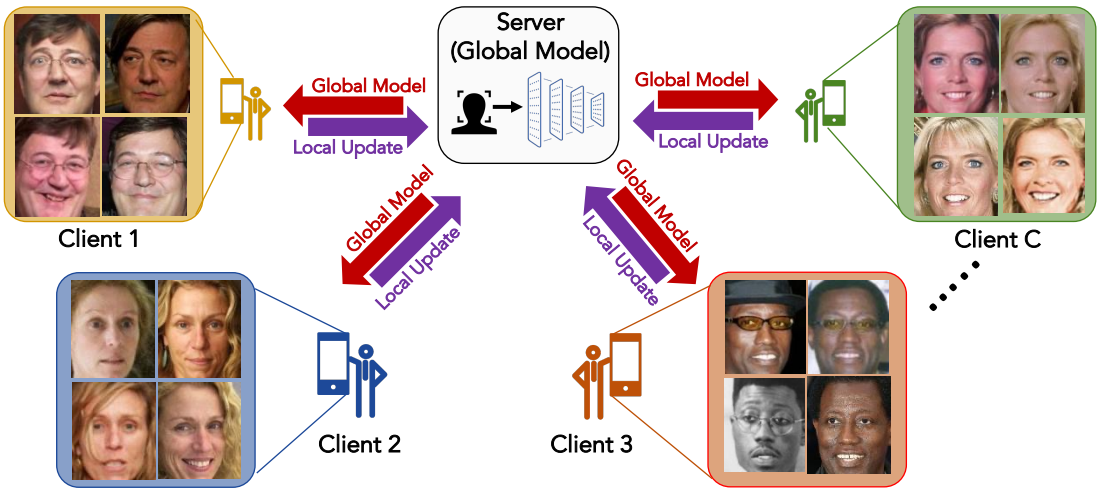


Figure 1: An overview of the federated setup for training face recognition models. We tackle the challenging but realistic scenario where each client is a mobile device with face images pertaining to only the user/owner of the device. In each communication round, the server sends the weights of the global model at that communication round to each of the participating client nodes who update the model based on their local training data. These local updates are transferred back to the server where they are aggregated to generate a global update. Through several rounds of communication between the server and the clients, a collaborative face recognition model can be learned in a privacy aware manner. Note that a client does not reveal its face images to the server or other clients.

systems in a distributed and privacy aware manner is now becoming extremely important. Specifically, we need to directly learn salient features from face images on end user devices such as mobile phones while protecting their data (i.e, the face images on a client device are never revealed).

Federated learning (FL) provides a distributed and privacy-aware framework to train machine learning models where multiple client nodes collaboratively learn without sharing their data with the server or with other clients. McMahan *et al*. [24] proposed a method where the weights of the updated models generated by the participating clients after local training are communicated to the server; the server then aggregates these weights to obtain the weights for the global model. Variants of FedAvg [24] such as FedProx [21] and Agnostic Federated Learning [25] have also been proposed that aim at alleviating issues such as bias towards different clients when learning on clients with heterogeneous data. However, these conventional FL algorithms cannot be directly applied to an automated face recognition pipeline, since, the participating clients are mobile devices that only contain face images of the device owners, and thereby impose significant challenges in establishing discrimination among different faces.

In an effort to facilitate distributed training of face recognition systems from face images available on mobile devices, we propose a framework, called *FedFace*, that learns an accurate face recognition model from multiple mobile devices in a collaborative manner without sending training face images outside of the device. Main contributions of our work are summarized as follows:

- The *FedFace* framework, for training face recognition systems in the federated setup to alleviate aggregating face datasets on a common server thereby facilitating data privacy. We tackle the challenging but realistic scenario where each of the participating clients has face images of only one identity.
- Our empirical results show that *FedFace* enhanced the performance of a pre-trained face recognition system, CosFace [29], by utilizing additional face data available on client nodes in a collaborative manner. *FedFace* employs a mean feature initialization scheme for the class embeddings and a spreadout regularizer to ensure that the class embeddings are well separated.
- *FedFace* is able to enhance the performance of a pre-trained face recognition system namely, CosFace, from a TAR of 81.43% to 83.79% on IJB-A @0.1% FAR and accuracy from 99.15% to 99.28% on LFW using the face images available on $1,000$ mobile devices in such a setup.

2. Related Work

Federated learning is a machine learning paradigm that enables training machine learning models collaboratively across multiple devices holding local data [24, 12, 30, 11, 21, 25]. In the FL setup, the device that manages the overall model training is called the server and the devices that store the datasets and propagate their trained weights to the server are called client nodes. FL framework ensure that the local data of a client is never shared to other client nodes or the server. Therefore, by design, the FL framework keeps the

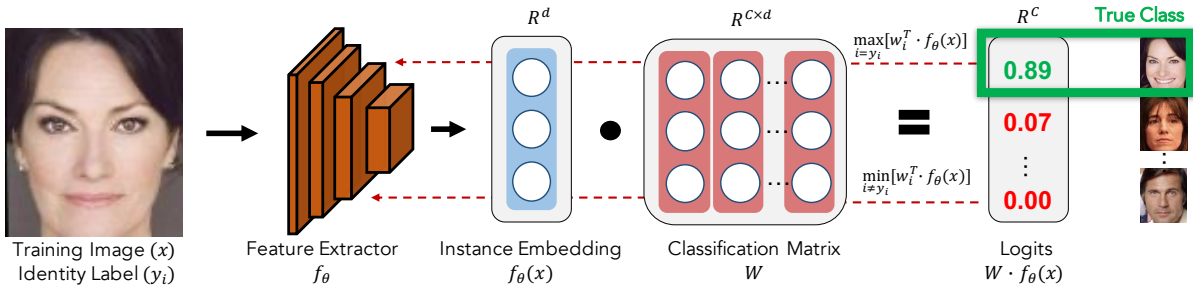


Figure 2: An overview of the training framework used by prevailing DNN-based AFR systems. An input x with a label y_i is passed through a feature extractor f_θ to obtain the feature vector $f_\theta(x)$. The feature vector is then multiplied with the classification matrix W to get the logits or the likelihood of x belonging to each of the C identities. We then maximize the similarity between the feature vector and the positive class embedding w_i and minimize the similarity between the feature vector and negative class embeddings (Red line indicates back-propagation of the loss through the model). In the FL setup, since each client does not have access to class embeddings of other clients/identities, the client cannot minimize the second term of the training objective.

data of the participating clients private and ensures security of the highly sensitive data. Several methods [18, 20] have been proposed in the literature to train deep learning models in the FL setting. Federated Averaging [24] (FedAvg) proposed by McMahan *et al*., which uses a weighted average of the local weight updates as the global weight update, is still one of the most widely adopted algorithms for training deep models in the FL setting.

FL has been widely adopted in a range of domains to train classification models in a privacy aware manner. Google uses FL in the Gboard mobile keyboard [15, 31, 13, 26] for next word prediction and other natural language processing tasks. FL is also being used extensively in health informatics [17, 28] as hospitals operate under strict privacy regulations [7] that prohibits their data from being shared with other organizations. However, its use in face recognition domain has been limited despite growing privacy concerns related to collection and sharing of face images.

The only FL framework that we could find in the face domain is by Shao *et al*. [27] who proposed a face presentation attack detection system, called FedPAD. They train a generalizable face presentation attack detection system by learning from different spoof types and data distributions stored at different sites in the FL setup. *However, to the best of our knowledge, no prior work has tackled the problem of learning a face recognition model under the FL setup.* More importantly, available FL algorithms cannot be directly applied to training face recognition models where the client nodes are mobile devices containing face images from only one identity. Yu *et al*. [33] proposed the Federated Averaging with Spreadout (FedAwS) algorithm that uses a geometric regularizer known as the Spreadout regularizer to tackle this problem for object recognition.

To the best of our knowledge, our work is the first to conduct a comprehensive evaluation of training face recognition models under the FL setup and to show that training with additional face images in a federated setting can help boost the performance of a pre-trained AFR system.

3. FedFace

3.1. Problem Setup

Face recognition systems are essentially embedding based classifiers *i.e*., both the classes (identities) and the input instance are embedded into a high dimensional Euclidean space. The similarity between the embedded input and the class embedding represents the likelihood of the instance belonging to that particular class. Formally, given an input face image x, the face feature extractor $f_\theta : X \rightarrow \mathcal{R}^d$ embeds the input instance into a d-dimensional feature vector $f_\theta(x)$. A fully connected layer with a softmax activation function parameterized by the class embedding matrix $W \in \mathcal{R}^{C\times d}$ (C represents the total number of identities), then maps this feature vector to the logits or scores for each identity. The k^{th} row of the class embedding matrix, w_k, is referred to as the class embedding of the k^{th} class. The logit for the k^{th} class is thus computed as $w_k^T \cdot f_\theta(x)$. A general overview of training a DNN-based AFR system is shown in Fig. 2. The training objective is as follows:

$$l(x,y) = \alpha \cdot (d(f_{\theta_t}(x), w_y))^2 + \beta \cdot \sum\nolimits_{c\neq y} (\max\{0, v - d(f_{\theta_t}(x), w_c)\}))^2, \quad (1)$$

where the first term defines the positive part (l_{pos}) and the second term defines the negative part (l_{neg}) of the loss function. d represents the distance function between the two input quantities. α and β are hyper-parameters denoting the weight of each term. v denotes the margin and c and y are the identity labels and the ground-truth identity label respectively. It essentially tries to accomplish the following:

- Maximize the similarity between the instance embedding $f_\theta(x)$ and the positive class embedding w_y where the instance x belongs to the identity y.
- Minimize the similarity between the instance embedding $f_\theta(x)$ and the negative class embeddings w_c for

all identities $c \neq y$, where the instance x belongs to the identity y.

At the face verification stage, we use the face feature extractor f_θ to extract the d-dimensional face representation. A distance metric (e.g., cosine similarity) between the two face features defines the similarity between the two face images. If the score is above a certain threshold τ, we say that the two faces belong to the same identity; otherwise we decide that the two face images do not belong to the same identity.

In the FL setup, we consider the realistic scenario where each of the client nodes actually correspond to individual user devices such as mobile phones which contain face images of only the owner of the device, rather than a central face dataset with multiple identities. Let us assume that there are C mobile devices or client nodes. The server contains a face recognition system f_{θ_0} trained on a publicly available dataset. The i^{th} client node contains n_i instances from one identity (the owner of the device). Therefore, the i^{th} client has access to the dataset S^i, where $S^i = (x^i_1, y^i), \ldots, (x^i_{n_i}, y^i)$. Our objective is to enhance the performance of the face recognition system available on the server by collaboratively learning from the data on each of the clients without sharing training face images outside of the device.

3.2. Federated Learning

FedAvg [24] is one of the most widely adopted FL algorithm for training deep models. In FedAvg [24], a server facilitates the collaborative training of the model as follows:

1. In the t^{th} communication round, the server sends the global model parameters θ_t and the class embedding matrix W_t to all the client nodes.

2. The i^{th} client updates the global model at iteration t based on the local data S^i and the loss function

$$L(f_{\theta_t}; S^i) = \frac{1}{n_i} \sum_{j \in [n_i]} l(f_{\theta_t}(x^i_j), y^i_j) \tag{2}$$

as follows

$$\theta^i_t = \theta_t - \eta \cdot \nabla_{\theta_t} L(f_{\theta_t}; S^i) \tag{3}$$

$$W^i_t = W_t - \eta \cdot \nabla_{W_t} L(f_{\theta_t}; S^i) \tag{4}$$

3. The server receives the updated model parameters (θ^i_t,W^i_t) from all the participating client nodes and updates the global model parameters by taking a weighted average of them as follows:

$$\theta_{t+1} = \frac{1}{n} \sum_{i \in [C]} n_i \cdot \theta^i_t \; ; W_{t+1} = \frac{1}{n} \sum_{i \in [C]} n_i \cdot W^i_t, \tag{5}$$

where n_i is the no. of training samples on the i^{th} client and n is the total no. of training samples across all client nodes.

4. Finally, the updated global parameters are transmitted to each of the clients and steps $2-4$ are repeated until convergence.

Our objective is to enhance the face recognition performance of a pre-trained AFR model at the server by utilizing the face images available *only* on end user devices (clients) such as mobile phones under the extreme case where each client has face images of only one identity. Conventional FL algorithms cannot be directly applied to such an instance because of two major reasons:

- In the t^{th} communication round, the server cannot transmit the entire classification matrix W_t to all users since the class embeddings contain sensitive information that can be potentially used to identify the user. Therefore, for the i^{th} user, the server only communicates the parameters of the feature extractor θ_t and the class embedding w^i_t associated with that user.

- The loss function (Eqn. 1) encourages the instance embedding to be similar to the positive class embedding (l_{pos}) while at the same time minimizing the similarity between the instance embedding and the negative class embeddings (l_{neg}). In the current setting, the i^{th} user only has access to face images of the i^{th} identity and its positive class embedding w_i. Therefore, it cannot compute l_{neg} while updating the model parameters on its local data as it does not have access to the negative class embeddings.

Optimizing only the positive part of the loss function (l_{pos}) would lead to a trivial solution where all the face images and the class embeddings collapse into a single point in the feature space resulting in sub-optimal performance.

3.3. Proposed Method

To tackle the aforementioned challenges, an extra level of optimization at the server level is needed to ensure that the class embeddings are well separated in the feature space. To achieve this goal, we leverage the Spreadout regularizer [33] to make sure that the embeddings of different identities are evenly distributed in the feature space.

The proposed *FedFace* framework is summarized in Algorithm 1. The server contains a pre-trained face extractor parametrized by θ_0 trained on a publicly available dataset or a dataset readily available at the server. This is a reasonable assumption to make because there are celebrity face datasets such as Celeb-A [22] etc. available in the public domain, which can be used by the server for pre-training the system. The i^{th} client/edge device contains a local face

dataset S_i with n_i face images each of which belongs to the identity i. Therefore, in total we have C clients and C identities (each client holding multiple face images of one identity). *FedFace* aims to enhance the performance of the pre-trained face feature extractor by collaboratively learning on the additional face images available on each of the clients.

The server initializes a face feature extractor with the parameters θ_0 and communicates this global model to each of the clients. In the first communication round, each of the clients need to initialize the class embedding w_0^i for themselves which will then be locally updated using their local data along with the parameters θ_t. Generally, the clients randomly initialize the class embedding for themselves from a Gaussian distribution. However, the class embeddings may lie outside the feature space of the pre-trained face extractor and thus, forcing the instance embeddings to be similar to the class embeddings can inhibit the already leaned embedding space by the pre-trained feature extractor and lead to inferior performance on general face recognition tasks. To alleviate this issue, *FedFace* utilizes the mean of the embeddings of all the instances available at client i extracted using the pre-trained feature extractor f_{θ_0} as the initialization of the class embeddings. Formally, if client i has a dataset $S^i = (x_1^i, y^i), \ldots, (x_{n_i}^i, y^i)$, the initialized class embedding for client i can be denoted as :

$$w_0^i = \frac{1}{n_i} \sum\nolimits_{j=1}^{n_i} f_{\theta_0}(x_j^i). \tag{6}$$

Since the mean feature is a linear combination of finite number of embeddings in f_{θ_0}'s feature space, it also lies in the same feature space and thus provides a better initialization point for the class embeddings than the randomly initialized ones. We normalize the class embeddings to constrain them to lie in a d-dimensional hypersphere, *i.e.*, $||w_t^i||_2^2 = 1$.

In the t^{th} communication round, the i^{th} client then updates the parameters θ_t and w_t^i by optimizing the positive loss function l_{pos} using the local data S^i. *FedFace* uses the squared hinge loss with cosine distance to define l_{pos} at each of the individual clients

$$l_{pos}(f_{\theta_t}(x), i) = \max\left(0, m - (w_t^i)^T f_{\theta_t}(x)\right)^2, \tag{7}$$

where m denotes the margin hyper-parameter and w_i represents the class embedding for the i^{th} client.

The updated values of the parameters θ and w_i are communicated back to the server by all the clients. The server then aggregates the values of θ from each of the individual clients by taking a weighted average of them similar to the FedAvg algorithm. Since each of the clients only transmits the updated value of its own class embedding w_t^i, the server concatenates the class embeddings from all the clients to define the class embedding matrix W_t.

$$W_t = [w_t^1, w_t^2, \ldots, w_t^C]^T. \tag{8}$$

Algorithm 1: FedFace training procedure.

Input : A pre-trained face feature extractor f_{θ_0} at the server and the local dataset S^i consisting of n_i face images belonging to the same identity i at the i^{th} client. In total, we have C clients and C identities with each client having multiple face images of one identity

Initialization: T is the total number of communication rounds, η is the learning rate at each client and λ is the weight multiplier for the spreadout loss

The server initializes a global face feature extractor with the parameter θ_0 ;

The i^{th} client initializes the class embedding w_0^i with the mean feature as follows

$w_0^i = \frac{1}{n_i} \sum\limits_{j=1}^{n_i} f_{\theta_0}(x_j^i)$

l_2-normalize the class embeddings;

for $t = 1, \ldots, T$ **do**

  The server communicates the current global parameters θ_t and w_t^i to the i^{th} client (except for the first round where w_0^i is initialized by the client itself);

  for *each of the clients* $i = 1, 2, \ldots, C$ **do**

    The client updates the model parameters based on the local data S^i

    $(\theta_{t+1}^i, w_{t+1}^i) \leftarrow (\theta_t^i, w_t^i) - \eta \nabla_{(\theta_t^i, w_t^i)} L_{pos}(S^i)$

    where $L_{pos}(S^i) = \frac{1}{n_i} \sum\limits_{j=1}^{n_i} l_{pos}(f_{\theta_t^i}(x), i)$;

    The i^{th} client sends the updated parameters $(\theta_{t+1}^i, w_{t+1}^i)$ back to the server

  end

  Server aggregates the updated parameters from all the clients

  $\theta_{t+1} \leftarrow \frac{1}{n} \sum\limits_{i \in [C]} n_i \cdot \theta_{t+1}^i$

  where n represents the total number of samples on all the clients

  $\hat{W}_{t+1} = [w_{t+1}^1, \ldots, w_{t+1}^C]$;

  Finally, the server employs the spreadout regularizer to separate the class embeddings from each other

  $W_{t+1} \leftarrow \hat{W_{t+1}} - \lambda \nabla_{\hat{W_{t+1}}} reg_{sp}(\hat{W_{t+1}})$

end

Output : Output θ_T

Finally, to ensure that the class embeddings are well separated in the embedding space and do not collapse into a single point, we use the spreadout regularizer inspired by the FedAwS algorithm. After aggregating the class embeddings from all the clients in each communication round, the server optimizes the spreadout regularizer which is defined as follows:

$$reg_{sp}(W_t) = \sum_{c \in [C]} \sum_{\hat{c} \neq c} (\max\{0, v - d(w_t^c, w_t^{\hat{c}})\})^2, \quad (9)$$

where v represents the margin.

The above steps are repeated for T communication rounds at the end of which the global model parameter θ_T for the face feature extractor are returned. See Algorithm 1 for the complete procedure.

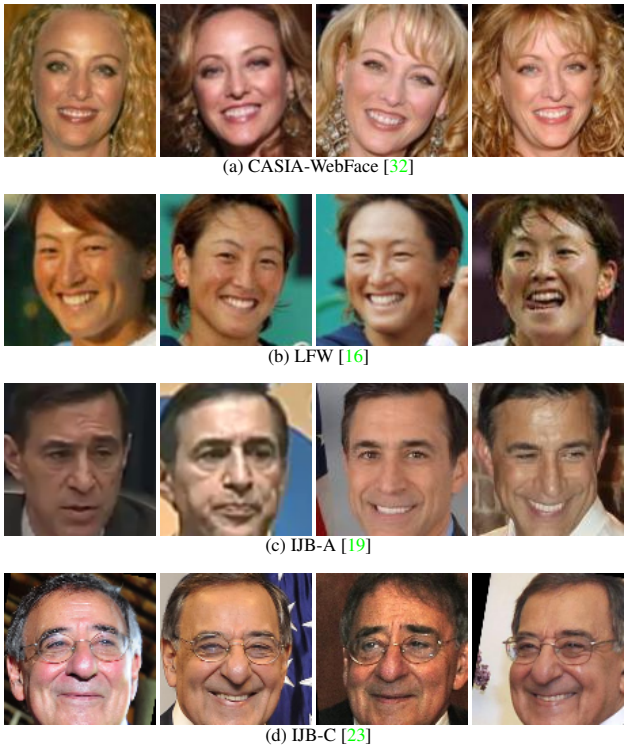
4. Experimental Results

4.1. Experimental Settings

Datasets : We train *FedFace* using face images in CASIA-WebFace [32] dataset. CASIA-WebFace comprises of $494,414$ images from $10,575$[1] different subjects. We randomly choose $10,000$ subjects from the CASIA-WebFace dataset as our training set. To test the performance of *FedFace* on Face Verification task, we evaluate on three different face benchmarks, namely LFW [16], IJB-A [19] and IJB-C [23]. LFW [16] contains $13,233$ face images of $5,749$ subjects. We evaluate *FedFace* on LFW under the standard LFW protocol. IJB-A [19] utilizes a template-based benchmark, containing $25,813$ faces images of 500 subjects. Each template includes a set of still photos or video frames. IJB-C [23] is an extension of IJB-A with $140,740$ faces images of $3,531$ subjects. Note that IJB-C is not a more challenging dataset than IJB-A and their performances are comparable in the literature.

Implementation : For the face feature extractor, we use a publicly available face matcher, CosFace [29] (64 layer), which outputs a 512-dimensional feature vector. For pre-training the global face recognition model on the server, we use an angular margin softmax loss with the scale hyper-parameter value of 30 and the margin hyper-parameter value of 10 similar to CosFace. For training in the FL setup at each individual client, we use a squared hinge loss with cosine distance with a margin of 0.9 as the positive loss function. We normalize both the instance embeddings and the class embeddings to make their norm as 1. We train the model for 200 communication rounds. We choose a learning rate of 0.001 to update the model locally on each of the clients. We use a small learning rate to make sure that training in the collaborative setting does not deviate much from the original embedding space. We use a sufficiently large weight for the spreadout regularizer $\lambda = 10$) to ensure that the class embeddings are well separated.

[1]We remove 84 subjects in CASIA-WebFace that overlap with LFW



(a) CASIA-WebFace [32]

(b) LFW [16]

(c) IJB-A [19]

(d) IJB-C [23]

Figure 3: Examples of face images from (a) CASIA-WebFace [32], (b) LFW [16], (c) IJB-A [19] and (d) IJB-C [23] datasets. Each row consists of face images of one identity.

4.2. Effect of number of clients

To evaluate the effect of the number of clients participating in the collaborative learning on the performance of the conventional FedAvg [24] algorithm, we divide our training set ($10,000$ subjects of CASIA-WebFace [32]) into a number of clients with equal number of subjects available for training at each client. One client scenario represents the traditional way of learning face recognition models where all the data is available centrally. We test on 4 (2500 subjects on each client), 8 ($1,250$ subjects on each client), 16 (625 subjects on each client), 64 (156 subjects on each client) and finally $10,000$ (1 subject on each client) client nodes. Fig. 4 represents the performance of FedAvg [24] on IJB-A [19] dataset under these settings. Note that since FedAvg uses the AM-Softmax loss function for training on the clients without the spreadout regularizer, it does not enforce the class embeddings to be well separated. Therefore, in the $10,000$ clients case, FedAvg leads to a trivial solution with the feature vectors for all the face images clustering around a single point deteriorating the performance significantly.

Table 1: Face Verification performance of *FedFace* on standard face recognition benchmarks LFW [16], IJB-A [19] and IJB-C [23].

Method	Training Data	LFW [16]	IJB-A [19]	IJB-C [23]
		LFW Accuracy(%)	TAR @ 0.1% FAR	TAR @ 0.1% FAR
Baseline	Centrally aggregated	99.15%	81.43%	84.78%
Fine-tuning baseline in a non-federated manner	Centrally aggregated	99.32%	84.18%	88.76%
Randomly Initialized class embeddings	Distributed	94.61%	70.13%	69.30%
FedAvg [24]	Distributed	68.88%	19.75%	14.04%
Proposed FedFace	**Distributed**	**99.28%**	**83.79%**	**88.21%**

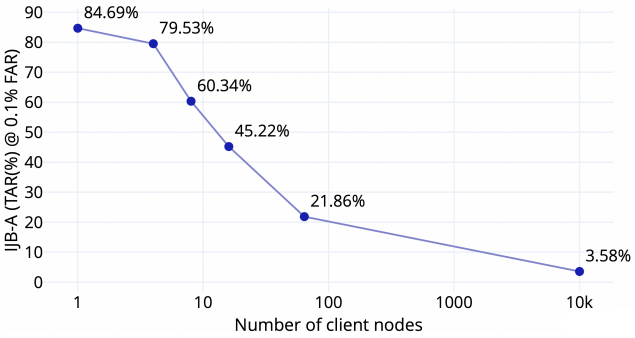


Figure 4: Effect of the number of clients on the FedAvg [24] algorithm. We divide the $10,000$ subjects in CASIA-WebFace [32] equally into different client nodes for training. One client node denotes the conventional (non-federated) way of training AFR systems while the $10k$ clients represents the problem we are tackling with face images of one identity per client. We evaluate on IJB-A [19]. Note that the x-axis is in log scale.

(a) (b)

Figure 5: Examples of two genuine pairs of face images from LFW dataset, (a) Saeb Erekat and (b) Sarah Hughes which failed to match using the pre-trained face feature extractor. After training on the additional $1,000$ subjects using *FedFace*, the two genuine pairs were correctly matched.

4.3. Performance of FedFace on Standard Face benchmarks

To evaluate the performance of *FedFace*, we divide our training set into two parts. $9,000$ of the $10,000$ subjects in the training set are used to pre-train a face recognition system on the server. This set simulates publicly available face datasets or face images available on the server for pre-training. The remaining $1,000$ subjects are distributed into $1,000$ clients with each client having access to all the face images of one subject. We evaluate *FedFace* in this setting on three different face benchmarks namely LFW [16], IJB-A [19] and IJB-C [23]. We compare the proposed *FedFace* (Section 3.3) with the following methods:

Baseline: Pre-training CosFace with face images of the $9,000$ subjects from CASIA-WebFace. This is the face recognition model that the server has access to initially.

Fine-tuning baseline in a non-federated manner: Centrally aggregating the face images of the $1,000$ clients and using them to boost the performance of the pre-trained face recognition model in a non-federated manner. This is the traditional way of training DNN based face recognition models but suffer from serious privacy concerns since the face images are shared to the server.

Randomly initialized class embeddings: As explained in Section 3.2, training with only the positive loss function (l_{pos}) can lead to a trivial solution where all the class embeddings and face feature vectors collapse into a single point in the embedding space. One way to counter this is to randomly initialize the class embeddings and fixing them during training. This prevents the class embeddings from collapsing into single point but does not explicitly force them to be well separated. since the face images are shared to the server.

FedAvg: The conventional FedAvg [24] algorithm where the AFR system is trained in the FL setup but with only the positive part of the loss function (l_{pos}).

Table 1 shows the comparison of *FedFace* with the above methods. *FedFace* is able to boost the performance of the pre-trained face recognition system by collaboratively learning from the additional face images available on each of the individual clients. It effectively bridged the gap between the non-federated setup and the federated setup of learning while ensuring that the face images available on each of the clients are kept private. Fig. 5 shows two genuine pairs of face images from the LFW dataset which failed to match using the pre-trained face feature extractor but after training on the additional $1,000$ subjects using FedFace, they were matched correctly.

4.4. Ablation Study

We ablate over the mean feature initialization scheme and the spreadout regularizer to study their individual effect on the performance of *FedFace*. The results of the ablation study are summarized in Table 2. Note that although, the mean feature initialization does not help significantly in boosting the final performance, it helps in faster convergence (200 communication rounds compared to 450 communication rounds with random initialization) and is therefore, extremely crucial for such a collaborative learn-

ing framework where the communication latency between the clients and the server is usually a critical bottleneck.

Table 2: Ablation Study

	LFW [16] LFW Accuracy	IJB-A [19] TAR @ 0.1% FAR	IJB-C [23] TAR @ 0.1% FAR
w/o mean feature initialization	99.28%	83.73%	88.12%
w/o Spreadout Regularizer	75.90%	34.16%	25.40%

5. Conclusion

In this paper, we tackle the problem of learning a collaborative face recognition system from face images available on a collection of mobile devices in a privacy aware manner. We propose a federated learning framework called *FedFace* to enhance the performance of a pre-trained face recognition system, CosFace, under the extreme case where each mobile device has face images of only one identity. We evaluate our approach on three standard face recognition benchmarks namely LFW, IJB-A and IJB-C. The proposed approach is able to enhance the performance of CosFace by effectively utilizing the additional data available on mobile devices without the face images ever leaving the device. In the future, we aim to extend our work to adapt a pre-trained AFR system to new domains such as selfie images using face images available on mobile devices.

Acknowledgement

This research was supported in part by ONR grant N00014-20-1-2382.

References

[1] Brainwash Dataset. https://exposing.ai/brainwash/.

[2] California consumer privacy act. https://bit.ly/2R8aD8j.

[3] Clearview ai uses your online photos to instantly id you. that's a problem, lawsuit says. https://lat.ms/3uqRn3Z.

[4] Duke MTMC Dataset. https://exposing.ai/duke_mtmc/.

[5] Facial recognition technology - privacy and accuracy issues related to commercial uses. https://bit.ly/2OkV4ZM.

[6] The general data protection regulation (eu). https://bit.ly/3t3BqjW.

[7] HIPAA privacy rule. https://bit.ly/31Outaw.

[8] Illinois personal information protection act. https://bit.ly/3cPJeR1.

[9] Microsoft quietly deletes largest public face recognition data set. https://on.ft.com/2PA595J.

[10] NIST FRVT report 2020. https://bit.ly/2Ps1gzX.

[11] S. Augenstein, H. B. McMahan, D. Ramage, S. Ramaswamy, P. Kairouz, M. Chen, R. Mathews, et al. Generative models for effective ML on private, decentralized datasets. *arXiv preprint arXiv:1911.06679*, 2019.

[12] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečnỳ, S. Mazzocchi, H. B. McMahan, et al. Towards federated learning at scale: System design. *arXiv preprint arXiv:1902.01046*, 2019.

[13] M. Chen, R. Mathews, T. Ouyang, and F. Beaufays. Federated learning of out-of-vocabulary words. *arXiv preprint arXiv:1903.10635*, 2019.

[14] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In *ECCV*, 2016.

[15] A. Hard, K. Rao, R. Mathews, S. Ramaswamy, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*, 2018.

[16] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report, University of Massachusetts, Amherst, October 2007.

[17] L. Huang, Y. Yin, Z. Fu, S. Zhang, H. Deng, and D. Liu. Loadaboost: Loss-based adaboost federated machine learning with reduced computational complexity on iid and non-iid intensive care data. *PLOS One*, 15(4), 2020.

[18] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

[19] B. F. Klare, B. Klein, E. Taborsky, A. Blanton, J. Cheney, K. Allen, P. Grother, A. Mah, and A. K. Jain. Pushing the frontiers of unconstrained face detection and recognition: Iarpa janus benchmark A. In *CVPR*, 2015.

[20] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3), 2020.

[21] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127*, 2018.

[22] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[23] B. Maze, J. Adams, J. A. Duncan, N. Kalka, T. Miller, C. Otto, A. K. Jain, W. T. Niggel, J. Anderson, J. Cheney, et al. Iarpa janus benchmark-C: Face dataset and protocol. In *IEEE ICB*, 2018.

[24] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, 2017.

[25] M. Mohri, G. Sivek, and A. T. Suresh. Agnostic federated learning. In *ICML*, 2019.

[26] S. Ramaswamy, R. Mathews, K. Rao, and F. Beaufays. Federated learning for emoji prediction in a mobile keyboard. *arXiv preprint arXiv:1906.04329*, 2019.

[27] R. Shao, P. Perera, P. C. Yuen, and V. M. Patel. Federated face anti-spoofing. *arXiv preprint arXiv:2005.14638*, 2020.

[28] M. J. Sheller, B. Edwards, G. A. Reina, J. Martin, S. Pati, A. Kotrotsou, M. Milchenko, W. Xu, D. Marcus, R. R. Colen, et al. Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data. *Nature*, 10(1), 2020.

[29] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu. Cosface: Large margin cosine loss for deep face recognition. In *CVPR*, 2018.

[30] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. Quek, and H. V. Poor. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security (TIFS)*, 15, 2020.

[31] T. Yang, G. Andrew, H. Eichner, H. Sun, W. Li, N. Kong, D. Ramage, and F. Beaufays. Applied federated learning: Improving google keyboard query suggestions. *arXiv preprint arXiv:1812.02903*, 2018.

[32] D. Yi, Z. Lei, S. Liao, and S. Z. Li. Learning face representation from scratch. *arXiv preprint arXiv:1411.7923*, 2014.

[33] F. Yu, A. S. Rawat, A. Menon, and S. Kumar. Federated learning with only positive labels. In *ICML*, 2020.